Supplementary for Surprisingly Simple Semi-Supervised Domain Adaptation with Pretraining and Consistency

Samarth Mishra[1]
samarthm@bu.edu
Kate Saenko[12]
saenko@bu.edu
Venkatesh Saligrama[1]
srv@bu.edu

[1] Boston University
[2] MIT-IBM Watson AI Lab

A PAC performance with different target shots.

In Figure 1, we plot the target accuracy of 4 methods on the *real* to *clipart* adaptation scenario of Office-Home, for different number of labelled target examples. The method "CR" represents the consistency regularization part of PAC, meaning it starts with an Imagenet pretrained backbone, same as S+T and MME [13]. We see that with its domain alignment approach, MME performs well at 0 shots. However, along with pretraining using rotation prediction, which has some alignment effect, PAC does not lag far behind. As the number of labelled examples increase, we see all methods enjoy a significant boost in performance, where the error has an exponential relation to the number of labelled examples as indicated by Eq. 1 (main paper). Since PAC and CR have better feature space clustering, *i.e.*, they have a higher inter-class divergence D, they see a bigger reduction in error.

B More questions

Can consistency regularization fix more errors than MME? Short answer : yes. In Section 5 of the main paper, we mentioned that consistency regularization, because of the perturbations it makes in image space, can fix errors of the kind that simple conditional entropy minimization, the way it is done in MME, cannot. We validate this hypothesis by training both methods from a randomly initialized feature extractor, where we expect initial features to have a much less meaningful neighborhood in feature space. In Table 1, we see a larger gap in the performance of MME starting from a pretrained vs a randomly initialized backbone, which tells us that consistency regularization can fix a lot more errors in the initial feature space than MME. Note that "Ours (CR)" method here does not include any rotation pretraining for this comparison.

Which perturbation technique is best for consistency? We compared three different image augmentation approaches : RandAugment [3] involves a list of 14 different augmenta-

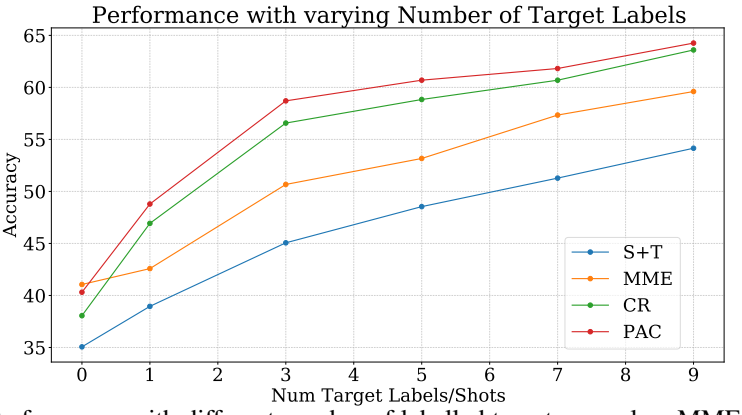

Figure 1: Performance with different number of labelled target examples. MME benefiting from domain alignment performs best at 0 shots. With more labelled examples, there is an exponential decrease in target error (Eq. 1), with PAC and CR benefiting most due to better target clustering, *i.e.*, high inter-class divergence D

Method	Imagenet pt.	Random init.
MME	51.2	26.9
Ours (CR)	54.1	40.0

Table 1: Comparison of MME and our consistency regularization approach on Imagenet pretrained backbone and randomly initialized backbone. Consistency regularization can fix more initial feature space errors than MME.

tion schemes like translations, rotations, shears, color/brightness enhancements etc., 2 out of which are chosen randomly anytime an image is augmented. We also evaluated color jittering, since common objects in our datasets are largely invariant to small changes in color. Finally we tried a combination of both, and found that this performed best for our method. Fig 2 shows the comparison of the final target accuracies achieved using an Alexnet backbone on the *real* to *clipart* adaptation scenario of Office-Home. Besides perturbations based on augmentation, we also evaluated adversarial image perturbation via virtual adversarial training (VAT) [10]. When using VAT, we found improvements over the simple "S+T" method (48.3% using VAT vs 44.6% without), but as seen from Fig 2, we found this was much lower than image augmentation approaches. This is quite likely because image augmentation imposes a more meaningful neighborhood on images where class labels do not change, while adversarial perturbation does not have this guarantee.

Can pretraining and consistency help other methods? An indication towards the affirmative is seen when we train MME with pretraining and consistency on the 3-shot *real* to *sketch* scenario of DomainNet using a Resnet-34 backbone. The results are shown in Table 2, where we can see that pretraining and consistency both individually help MME's performance, and their combination helps it the most.

It was explained how pretraining improves initial feature space, but prior work has also used "pretext" tasks like rotation prediction alongside classification for training [16, 19]. How does pretraining compare to that? A comparison of this can be found in table 3, which reports the 3-shot SSDA target accuracies of the two methods on the DomainNet

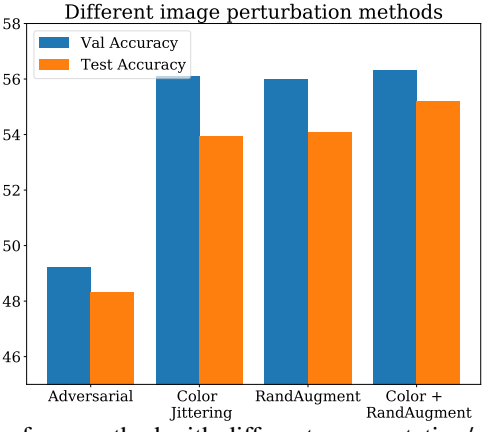

Figure 2: Performance of our method with different augmentation/perturbation methods on real to clipart adaptation of Office-Home. Adversarial perturbation helps, but not as much as image augmentation approaches do. A combination of color jittering and RandAugment performed the best.

Rot^n	CR	Accuracy
		61.9
✓		65.8
	✓	70.4
✓	✓	71.5

Table 2: Pretraining and consistency with MME.

dataset. As can be seen, pretraining using rotation prediction provides more of a performance benefit as compared to using rotation prediction as an auxiliary task like [16, 19]. The latter can help regularize final target classifier training, but likely does not have the benefits that pretraining provides the method via a better initial feature space for training.

Method	C2S	P2C	P2R	R2C	R2P	R2S	S2P	Mean
S+T + Rot^n pred	54.7	59.5	74.1	60.4	62.3	51.8	59.2	60.3
S+T (Rot^n pred pretrained backbone)	59.1	65.3	74.0	64.1	63.9	56.1	61.7	63.5

Table 3: Comparison of rotation prediction for pretraining vs as an auxilliary training task using target accuracies on 3-shot SSDA on different scenarios of DomainNet.

What if pretraining uses rotation prediction only on target? We train the backbone only on target domain data for pretraining with rotation prediction, and then train it like PAC using consistency regularization. On the 3-shot *real* to *clipart* SSDA scenario of Office-Home using an Alexnet backbone, this achieves a final target accuracy of 57.5% compared to 58.9% of PAC. This is indicative of target-only rotation prediction helping the initial feature extractor, but not as much as in the case when source domain data is used along with it.

How big is the role of source domain data in final target performance? To see this, we

Rot^n	CR	Accuracy (with source)	Accuracy (only target)
	✓	56.6	35.5
✓	✓	58.9	36.7

Table 4: Ablating source domain information.

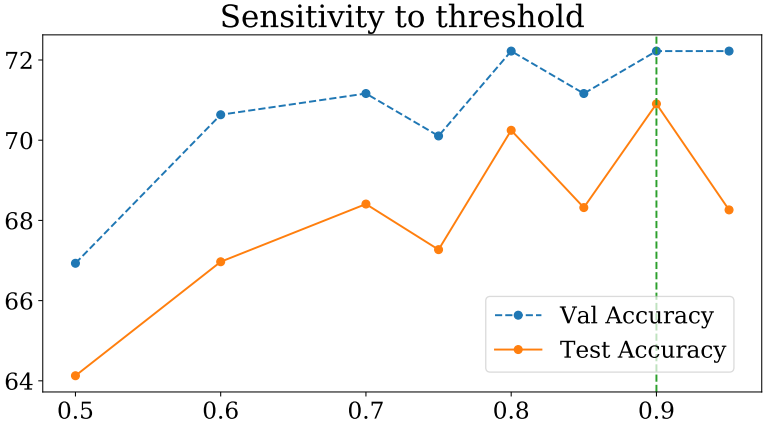

Figure 3: Sensitivity of our method to different thresholds used for consistency regularization. Accuracies reported are on the 3-shot *real* to *sketch* scenario of DomainNet using a Resnet-34 backbone.

train our method with no access to source domain data. This is similar to the semi-supervised learning problem. Target accuracy with only 3 labelled target examples and access to all other unlabelled examples, on the *clipart* domain of Office-Home using an Alexnet backbone, are in the last column of Table 4. For reference, the accuracies of our method with source domain data from the *real* domain (*i.e.* R2C adaptation scenario) are provided in the 3rd column.

C PAC sensitivity to confidence threshold

Our consistency regularization approach uses soft targets based on outputs of the classifier only in cases where the confidence of labelling is high. In Fig 3, we compare the sensitivity of our method to this threshold. We see that higher confidence thresholds up to 0.9 help final target classification performance.

D Results on Office and Office-Home

Office-Home [17] is a dataset with 65 categories of objects found in typical office and home environments. It has 4 different visual domains (Art, Clipart, Product, and Real), and we evaluate our methods on all 12 different adaptation scenarios. The 4 domains have close to 3800 images on average. Office [12] dataset has objects of 31 different categories in

Network	Method	D to A		W to A	
		1-shot	3-shot	1-shot	3-shot
Alexnet	S+T	50.0	62.4	50.4	61.2
	DANN	54.5	65.2	57.0	64.4
	ADR	50.9	61.4	50.2	61.2
	CDAN	48.5	61.4	50.2	60.3
	ENT	50.0	66.2	50.7	64.0
	MME	55.8	67.8	57.2	67.3
	APE	-	69.0	-	67.6
	BiAT	54.6	68.5	57.9	68.2
	CDAC	**63.4**	**70.1**	**62.8**	**70.0**
	PAC	54.7	66.3	53.6	65.1
VGG	S+T	68.2	73.3	69.2	73.2
	DANN	70.4	74.6	69.3	75.4
	ADR	69.2	74.1	69.7	73.3
	CDAN	64.4	71.4	65.9	74.4
	ENT	72.1	75.1	69.1	75.4
	MME	**73.6**	**77.6**	**73.1**	**76.3**
	PAC	72.4	75.6	70.2	76.0

Table 5: Results on Office. We evaluate using the two scenarios where the target domain is *amazon*

3 different domains—amazon, webcam and dslr, with approx. 2800, 800 and 500 images respectively. Following [13] we evaluated only on the 2 cases with amazon as the target domain, since the other two domains have a lot fewer images.

Table 6 shows the results of PAC on the different scenarios of Office-Home, the average accuracy over all these scenarios was also reported in Table 3 in the main paper. Table 5 shows the accuracy of PAC on two scenarios of Office. We see that PAC performs comparably to state of the art. It lags behind a little in the 1-shot scenarios as compared to 3-shot ones.

E Experiment details

All our experiments were implemented in PyTorch [11] using W&B [1] for managing experiments.

E.1 PAC experiments

We used three different backbones for evaluation in different experiments—Alexnet [9], VGG-16 [15] and Resnet-34 [7]. Our backbones before being trained using the rotation prediction task, are pretrained on the Imagenet [4] dataset, same as other methods used for comparison. While using an Alexnet or VGG-16 feature extractor, we use 1 fully connected layer as the classifier, and while using the Resnet-34 backbone, we use a 2-layer MLP with 512 intermediate nodes. The classifier C uses a temperature parameter set to 0.05 to sharpen the distribution it outputs using a softmax. For consistency regularization, the confidence threshold τ was set to 0.9 across all experiments, having validated on the *real* to *sketch* scenario of DomainNet.

Network	Method	R to C	R to P	R to A	P to R	P to C	P to A	A to P	A to C	A to R	C to R	C to A	C to P	Mean
One-shot														
Alexnet	S+T	37.5	63.1	44.8	54.3	31.7	31.5	48.8	31.1	53.3	48.5	33.9	50.8	44.1
	DANN	42.5	64.2	45.1	56.4	36.6	32.7	43.5	34.4	51.9	51.0	33.8	49.4	45.1
	ADR	37.8	63.5	45.4	53.5	32.5	32.2	49.5	31.8	53.4	49.7	34.2	50.4	44.5
	CDAN	36.1	62.3	42.2	52.7	28.0	27.8	48.7	28.0	51.3	41.0	26.8	49.9	41.2
	ENT	26.8	65.8	45.8	56.3	23.5	21.9	47.4	22.1	53.4	30.8	18.1	53.6	38.8
	MME	42.0	69.6	**48.3**	**58.7**	37.8	**34.9**	52.5	**36.4**	**57.0**	**54.1**	**39.5**	**59.1**	49.2
	BiAT	-	-	-	-	-	-	-	-	-	-	-	-	**49.6**
	PAC	**49.6**	**69.8**	45.9	57.5	**42.5**	30.4	**53.1**	35.8	51.9	48.2	26.0	57.6	47.4
VGG	S+T	39.5	75.3	61.2	71.6	37.0	52.0	63.6	37.5	69.5	64.5	51.4	65.9	57.4
	DANN	52.0	75.7	62.7	72.7	45.9	51.3	64.3	44.4	68.9	64.2	52.3	65.3	60.0
	ADR	39.7	76.2	60.2	71.8	37.2	51.4	63.9	39.0	68.7	64.8	50.0	65.2	57.3
	CDAN	43.3	75.7	60.9	69.6	37.4	44.5	67.7	39.8	64.8	58.7	41.6	66.2	55.9
	ENT	23.7	77.5	64.0	74.6	21.3	44.6	66.0	22.4	70.6	62.1	25.1	67.7	51.6
	MME	49.1	78.7	**65.1**	**74.4**	46.2	**56.0**	68.6	**45.8**	**72.2**	**68.0**	**57.5**	**71.3**	**62.7**
	PAC	**56.4**	**78.8**	64.6	73.1	**54.7**	55.3	**69.8**	43.5	69.5	65.3	45.3	69.6	62.2
Three-shot														
Alexnet	S+T	44.6	66.7	47.7	57.8	44.4	36.1	57.6	38.8	57.0	54.3	37.5	57.9	50.0
	DANN	47.2	66.7	46.6	58.1	44.4	36.1	57.2	39.8	56.6	54.3	38.6	57.9	50.3
	ADR	45.0	66.2	46.9	57.3	38.9	36.3	57.5	40.0	57.8	53.4	37.3	57.7	49.5
	CDAN	41.8	69.9	43.2	53.6	35.8	32.0	56.3	34.5	53.5	49.3	27.9	56.2	46.2
	ENT	44.9	70.4	47.1	60.3	41.2	34.6	60.7	37.8	60.5	58.0	31.8	63.4	50.9
	MME	51.2	73.0	50.3	61.6	47.2	40.7	63.9	43.8	61.4	59.9	44.7	64.7	55.2
	APE	51.9	74.6	51.2	61.6	47.9	**42.1**	**65.5**	44.5	60.9	58.1	44.3	64.8	55.6
	BiAT	-	-	-	-	-	-	-	-	-	-	-	-	56.4
	CDAC	54.9	**75.8**	**51.8**	**64.3**	51.3	**43.6**	65.1	47.5	**63.1**	**63.0**	**44.9**	**65.6**	**56.8**
	PAC	**58.9**	72.4	47.5	61.9	**53.2**	39.6	63.8	**49.9**	60.0	54.5	36.3	64.8	55.2
VGG	S+T	49.6	78.6	63.6	72.7	47.2	55.9	69.4	47.5	73.4	69.7	56.2	70.4	62.9
	DANN	56.1	77.9	63.7	73.6	52.4	56.3	69.5	50.0	72.3	68.7	56.4	69.8	63.9
	ADR	49.0	78.1	62.8	73.6	47.8	55.8	69.9	49.3	73.3	69.3	56.3	71.4	63.1
	CDAN	50.2	80.9	62.1	70.8	45.1	50.3	74.7	46.0	71.4	65.9	52.9	71.2	61.8
	ENT	48.3	81.6	65.5	76.6	46.8	56.9	73.0	44.8	**75.3**	**72.9**	59.1	**77.0**	64.8
	MME	56.9	**82.9**	65.7	**76.7**	53.6	**59.2**	75.7	54.9	**75.3**	**72.9**	**61.1**	76.3	67.6
	PAC	**63.5**	82.3	**66.8**	75.8	**58.6**	57.1	**75.9**	**56.7**	72.2	70.5	57.7	75.3	**67.7**

Table 6: Results on all adaptation scenarios of Office-Home.

Same as [13], we train the models using minibatch-SGD, with s source examples, s labelled target examples and $2s$ unlabelled target examples that the learner "sees" at each training step. $s = 24$ for the VGG and Resnet backbones, while $s = 32$ for Alexnet. The SGD optimizer used a momentum parameter 0.9 and a weight decay (coefficient of ℓ_2 regularizer on parameter norm) of 0.0005. For all experiments, the parameters of the backbone are updated with a learning rate of 0.001, while the parameters of the classifier are updated with a learning rate 0.01. Both of these are decayed as training progresses using a decay schedule similar to [5]. Learning rate at step i (η_i) is set as below:

$$\eta_i = \frac{\eta_0}{(1+0.0001 \times i)^{0.75}}$$

For experiments on the Office and Office-Home dataset, we trained PAC using both an Alexnet and a VGG-16 backbone, and the models were trained for 10000 steps with the stopping point chosen using best validation accuracy.

For the experiments on DomainNet, we use both Alexnet and Resnet-34 backbones, while for VisDA-17, we use only Resnet-34. All models in these experiments were trained for 50000 steps, using validation accuracy for determining the best stopping point.

E.2 Pretraining

As mentioned above, we pretrain our models for rotation prediction starting from Imagenet pretrained weights. A comparison of PAC with a backbone trained with rotation prediction starting from imagenet pretraining (final target accuracy = 58.9%) vs one that does not use any imagenet pretraining (final target accuracy = 43.7%), revealed that there is important feature space information in imagenet pretrained weights that rotation prediction could not capture on its own. This comparison was done using an Alexnet on the *real* to *clipart* adaptation scenario of Office-Home.

Following Gidaris *et al*. [6], we trained the model on all 4 rotations of a single image in each minibatch. Each minibatch contained s images each from source and target domains, which translates to $4s$ images considering all rotations. The Alexnet backbones are trained using a learning rate of 0.01 and $s = 128$. The Resnet-34 and VGG backbones are both trained using $s = 16$ and a learning rate of 0.001. We found that beyond a certain point early on in training, the number of steps of training for rotation prediction did not make a big difference to the final task accuracy, and finally the chosen number of training steps was 4000 for Alexnet, 2000 for VGG-16 and 5000 for Resnet-34 backbones.

E.3 Other Experiments

MoCo pretraining. Using the Alexnet backbone, we trained momentum contrast [8] for 5000 training steps, where in each step the model saw 32 images each from the *real* and the *clipart* domains of Office-Home. The queue length used for MoCo was 4096 and the momentum parameter was 0.999.

SimSiam pretraining. Using the Alexnet backbone, we trained SimSiam [7] for 200 epochs (or 6800 training steps) on a mix of the source (*real*) and target (*clipart*) sets of Office-Home, with a batch size of 256.

Virtual Adversarial Training. For adding a VAT criterion to our model, we closely followed the VAT criterion in VADA [14]. We used a radius of 3.5 for adversarial perturbations

and a coefficient of 0.01 for the VAT criterion, which is the KL divergence between the outputs of the perturbed and the unperturbed input from the target domain.

Empirical Bhattacharyya Distance Estimate. We use this estimate to compare target domain inter-class separation in Table 5 of the main paper. For computing an approximation, we made the assumption that features for each class in the target domain are distributed as gaussians with identity covariance and used the closed form Bhattacharyya Distance (BD) between two multivariate gaussians [18]. The estimate then reduces to:

$$BD = \frac{1}{\binom{K}{2}} \sum_{\substack{i,j \in [K] \\ i \neq j}} \frac{1}{8} \left|\left|\mu_i - \mu_j\right|\right|_2^2$$

where μ_i is the mean of class i features of images in the target domain ($\mathcal{D}_t \cup \mathcal{D}_u$).

References

[1] Lukas Biewald. Experiment tracking with weights and biases, 2020.

[2] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[3] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[5] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[6] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[10] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[12] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.

[13] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8050–8058, 2019.

[14] Rui Shu, Hung Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. In *International Conference on Learning Representations*, 2018.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015.

[16] Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019.

[17] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *(IEEE) Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[18] Wikipedia contributors. Bhattacharyya distance, 2020. URL https://en.wikipedia.org/wiki/Bhattacharyya_distance.

[19] Jiaolong Xu, Liang Xiao, and Antonio M López. Self-supervised domain adaptation for computer vision tasks. *IEEE Access*, 7:156694–156706, 2019.